Fast conformational clustering of extensive molecular dynamics simulation data

Cite as: J. Chem. Phys. 158, 144109 (2023); doi: 10.1063/5.0142797
Submitted: 17 January 2023 • Accepted: 20 March 2023 •
Published Online: 12 April 2023

Simon Hunkler,[1] Kay Diederichs,[1] Oleksandra Kukharenko,[2,a)] and Christine Peter[1,b)]

AFFILIATIONS

[1] Department of Chemistry, University of Konstanz, Konstanz, Germany
[2] Theory Department, Max Planck Institute for Polymer Research, Mainz, Germany

Note: This paper is part of the JCP Special Topic on Machine Learning Hits Molecular Simulations.
[a)]**Electronic mail:** kukharenko@mpip-mainz.mpg.de
[b)]**Author to whom correspondence should be addressed:** christine.peter@uni-konstanz.de

ABSTRACT

We present an unsupervised data processing workflow that is specifically designed to obtain a fast conformational clustering of long molecular dynamics simulation trajectories. In this approach, we combine two dimensionality reduction algorithms (cc_analysis and encodermap) with a density-based spatial clustering algorithm (hierarchical density-based spatial clustering of applications with noise). The proposed scheme benefits from the strengths of the three algorithms while avoiding most of the drawbacks of the individual methods. Here, the cc_analysis algorithm is applied for the first time to molecular simulation data. The encodermap algorithm complements cc_analysis by providing an efficient way to process and assign large amounts of data to clusters. The main goal of the procedure is to maximize the number of assigned frames of a given trajectory while keeping a clear conformational identity of the clusters that are found. In practice, we achieve this by using an iterative clustering approach and a tunable root-mean-square-deviation-based criterion in the final cluster assignment. This allows us to find clusters of different densities and different degrees of structural identity. With the help of four protein systems, we illustrate the capability and performance of this clustering workflow: wild-type and thermostable mutant of the Trp-cage protein (TC5b and TC10b), NTL9, and Protein B. Each of these test systems poses their individual challenges to the scheme, which, in total, give a nice overview of the advantages and potential difficulties that can arise when using the proposed method.

© *2023 Author(s). All article content, except where otherwise noted, is licensed under a Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).* https://doi.org/10.1063/5.0142797

Downloaded from http://pubs.aip.org/aip/jcp/article-pdf/doi/10.1063/5.0142797/16814539/144109_1_5.0142797.pdf

I. INTRODUCTION

With the ever-growing power of computers over the past decades, researchers in the field of molecular dynamics (MD) have obtained access to increasingly long trajectories and thereby to increasingly large amounts of data. The introduction of supercomputers that are specifically designed to generate MD trajectories (Anton[1] and Anton 2[2]) is only the latest high point in this development. Furthermore, new sampling methods[3,4] and distributed computing projects, such as Folding@home,[5] have contributed to a massive increase in the generated simulation trajectories. With this increasing amount of data, it is essential to have powerful analysis tools to process and understand the underlying systems and processes.

There is a rapid increase in the application of unsupervised machine learning methods to analyze molecular simulation data. Two of the most used families of analysis techniques are clustering and dimensionality reduction (DR) algorithms. They help to find low-dimensional subspaces in which important aspects of the original data are preserved and to group the data based on a given similarity measure/metric and thereby gain a better overview and understanding. In practice, most of the times, clustering and DR methods are used in combination. The DR algorithms can be roughly divided into linear methods [the most known are principal component analysis (PCA)[6,7] and time-lagged independent component analysis (TICA)[8,9]], nonlinear methods [kernel and nonlinear PCA, multidimensional scaling (MDS),[10,11] and MDS-based methods such as sketch-map,[12] isomap,[13] diffusion maps,[14,15] or UMAP[16]], and autoencoder-based approaches (such as encodermap,[17,18] time-autoencoders,[19] variational autoencoders,[20] and Gaussian mixture variational autoencoders[21]). In terms of clustering algorithms, there is again a wide range of different methods:

K-means,[22,23] spectral-clustering,[24] DBSCAN,[25] density-peak clustering,[26] CNN-clustering,[27] root-mean-square deviation (RMSD) based clustering,[28] neural network-based VAMPnets,[29] etc. For a comprehensive overview of unsupervised ML methods commonly used to analyze MD simulation data, we refer the reader to Ref. 30.

Even from this incomplete list of available methods, it should become obvious that there are a lot of different clustering and DR methods. All these methods have their individual strengths and weaknesses. However, there are still open challenges in the successful usage of the listed methods for processing simulation data with a high spatial and temporal resolution. This is connected to either the proper choice of hyperparameters (such as the number of dimensions for DR methods, the number of expected states for some clustering algorithms, neural-network architectures, different cutoffs, and correlation times), expensive optimization steps, or the amount of data that could be processed simultaneously. In this work, we present a data processing scheme that combines three different algorithms into one workflow to create a powerful clustering machinery. It tackles a number of the mentioned challenges as it has a way to define an appropriate lower dimensionality of the data, does not require a priori information about the expected number of states, and is fast in processing extensive MD trajectories with both a very high dimensionality and a large number of observations. It is specifically designed to find conformational clusters in long molecular simulation data (Fig. 1).

We use two different DR algorithms, namely an algorithm called "cc_analysis" and the encodermap algorithm. The cc_analysis method belongs to the family of MDS-based techniques and was first introduced for the analysis of crystallographic data.[31,32] Here, it is used for the first time for projecting data of protein conformations. The dimensionality of the cc_analysis-space that is typically required is more than two (10–40 for the systems shown in this work), and the amount of data that can be efficiently projected simultaneously is limited by the available memory (about 50 000 frames for a 72 GB workstation). To process much longer trajectories and to obtain a two-dimensional representation, we use the second DR algorithm—encodermap.[33] Its loss function, however, consists of two parts: the autoencoder loss and a MDS-like distance loss, which introduces interpretability to the resulting 2D representation. Moreover, once the encodermap network is trained, the encoder function can be used to project data into the 2D map in an extremely efficient way. We use encodermap to project data into 2D and for a fast assignment of the additional members to the clusters defined in the cc_analysis space. Finally, we use the HDBSCAN (hierarchical density-based spatial clustering of applications with noise) algorithm[34] to cluster the data in the cc_analysis space and then visualize the resulting clusters in the 2D encodermap space. HDBSCAN is a combination of density and hierarchical clustering that can work efficiently with clusters of varying densities, ignores sparse regions, and requires a minimum number of hyperparameters. We apply it in a non-classical iterative way with varying RMSD-cutoffs to extract the protein conformations of different similarities.

The clustering workflow that we describe in this work combines the three before-mentioned algorithms to leverage their different strengths while avoiding the drawbacks of the individual methods. It thereby serves as a new route to extract a conformational clustering from large MD data. The clusters that are identified using this workflow are structurally highly consistent and can be used in various downstream analyses, e.g., kinetic model building, or for the initiation or evaluation of enhanced sampling techniques or to simply get an overview of the conformational variety in any given dataset. Subsequently, we will show how the scheme performs on long MD trajectories of wild-type and mutated Trp-cage with native and misfolded meta-stable states (208 and 3.2 μs long simulations); really extensive simulations of NTL9 (1877 μs); and Protein B, where only a small percent of the simulation data (5%) is in the folded state (104 μs).

II. METHODS

A. cc_analysis

For dimensionality reduction, we use the cc_analysis algorithm introduced in Refs. 31 and 32. This algorithm was originally developed to analyze crystallographic data, where the presence of noise and missing observations pose a challenge to data processing in certain experimental situations. The method separates the inter-dataset influences of random errors from those arising from systematic differences and reveals the relations between high-dimensional input features by representing them as vectors in a low-dimensional space. Due to this property, we expected it to be highly applicable to protein simulation data, where one seeks to ignore the differences arising from random fluctuations and to separate the conformations based

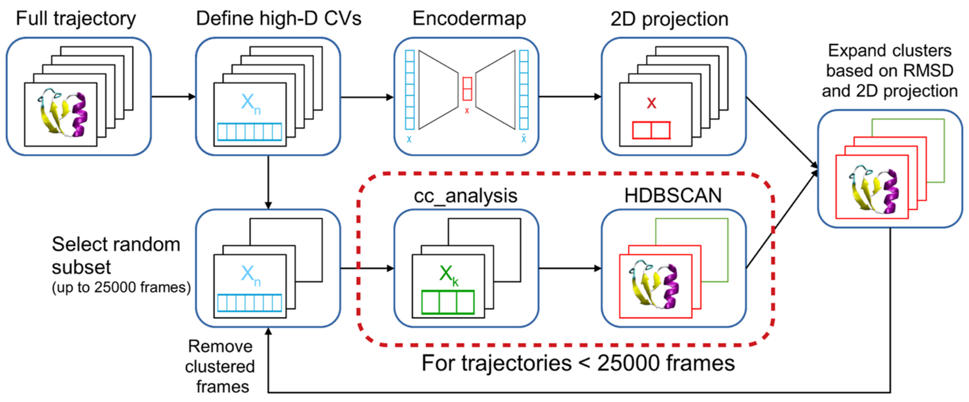


FIG. 1. Data processing routine presented in this article.

Downloaded from http://pubs.aip.org/aip/jcp/article-pdf/doi/10.1063/5.0142797/16814539/144109_1_5.0142797.pdf

© Author(s) 2023

on systematic differences. In the course of the manuscript, we show that this assumption proved to be correct.

The cc_analysis algorithm belongs to the family of MDS methods.[10] Its main distinction is that it minimizes the sum of squared differences between the Pearson correlation coefficients of pairs of high-dimensional descriptors and the scalar product of the low-dimensional vectors representing them [see Eq. (1)]. The procedure places the vectors into a unit sphere within a low-dimensional space. The systematic differences between the high-dimensional features lead to differences in the angular directions of the vectors representing them, and purely random differences in data points lead to different vector lengths in the same angular direction. The algorithm minimizes, e.g., iteratively using L-BFGS,[35] the expression

$$\Phi(\mathbf{x}) = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} (r_{ij} - x_i \cdot x_j)^2 \tag{1}$$

as a function of $\mathbf{x}$, the column vector of the N low-dimensional vectors $\{x_k\}$. Here, r_{ij} is the correlation coefficient between descriptors i and j in the high-dimensional space, $x_i \cdot x_j$ denotes the dot product of the unit vectors x_i and x_j representing the data in the low-dimensional space, and N is the number of observations, e.g., protein conformations. The output of cc_analysis is the N low-dimensional vectors $\{x_k\}$ and the eigenvalues of the $\mathbf{x}\mathbf{x}^T$ matrix.

To understand why this is a sensible approach, one can think about obtaining the least squares solution of Eq. (1) algebraically by the eigenanalysis of the matrix $\mathbf{r} = \{r_{ij}\}$. In that case, one would have to solve

$$\mathbf{x}\mathbf{x}^T = \mathbf{r},$$

where $\mathbf{r}$ is the matrix of the correlation coefficients r_{ij}. The n strongest eigenvalue/eigenvector pairs (eigenvectors corresponding to the largest eigenvalues) could then be used to reconstruct the N vectors x_i, which are located in an n-dimensional unit sphere. The systematic differences between the input data are thereby shown by the different angular directions in this low-dimensional sphere. This approximation is meaningful because, in general, the Pearson correlation coefficient can be written as a dot product between two vectors (after the subtraction of the mean and division by the standard deviation to scale the vectors to unit length) and is equal to the cosine of the angle between them. Hence, in an ideal scenario, $\sum_{i,j}^{N} x_i \cdot x_j$ can exactly reproduce the high-dimensional correlation coefficient matrix and $\Phi(\mathbf{x})$ in Eq. (1) would be equal to zero.

The length of the vectors is less important than the angle between them. The latter has an inherent meaning: two high-dimensional feature vectors with a correlation coefficient of zero between them would be projected to unit vectors at 90° angles with respect to the origin, and two feature vectors with a correlation coefficient of one would have a corresponding angle of zero degrees.

Despite the generality of the cc_analysis approach, by now, it has only been applied to crystallographic data[36,37] and protein sequence grouping.[38] Here, we present a first application of cc_analysis for protein simulation data analysis.

B. encodermap

To accelerate the processing of large datasets, e.g., from extensive simulations, in addition to cc_analysis, we make use of one more dimensionality reduction technique—encodermap. It was developed by Lemke and Peter[33] and is used here for the fast assignment of data points to clusters, as will be explained in detail in Sec. II D. The method combines the advantages of a neural network autoencoder[17] with a MDS contribution, here the loss function from the sketch-map algorithm[12] (Fig. 2). This combination is exceptionally efficient for projecting large simulation data to the two-dimensional representations: the sketch-map loss function allows us to concentrate only on relevant dissimilarities between conformations (ignoring thermal fluctuations and coping with the large dissimilarity values caused by the data's high dimensionality). Furthermore, the autoencoder approach allows us to avoid the complex minimization steps of the sketch-map projection and to process huge amounts of data in a very short time.

The encodermap loss function $L_{encodermap}$ [Eq. (2)] is a weighted sum of the autoencoder loss L_{auto} [Eq. (3)] and the sketch-map loss function L_{sketch} [Eq. (4)], which emphasizes mid-range distances by transforming all distances via a sigmoid function [Eq. (5)],

$$L_{encodermap} = k_a L_{auto} + k_s L_{sketch} + Reg, \tag{2}$$

$$L_{auto} = \frac{1}{N}\sum_{i=1}^{N} D(X_i, \tilde{X}_i), \tag{3}$$

$$L_{sketch} = \frac{1}{N}\sum_{i\neq j}^{N} [SIG_h(D(X_i, X_j)) - SIG_l(D(x_i, x_j))]^2, \tag{4}$$

where k_a and k_s are adjustable weights; Reg is a regularization used to prevent overfitting; N is the number of data points to be projected; $D(\cdot,\cdot)$ is the distance between points; X is the high-dimensional input; x is the low-dimensional projection (the bottleneck layer); and SIG_h and SIG_l are sigmoid functions of the form shown in Eq. (5),

$$\mathrm{SIG}_{\sigma,a,b}(D) = 1 - \left(1 + \left(2^{\frac{a}{b}} - 1\right)\left(\frac{D}{\sigma}\right)^{a}\right)^{-\frac{b}{a}}, \tag{5}$$

where a, b, and σ are parameters defining which distances to preserve.

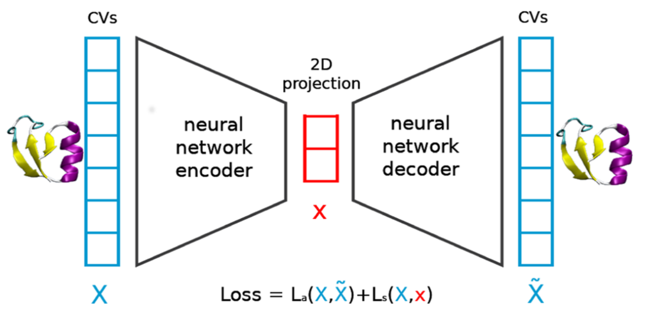


FIG. 2. Schematic description of encodermap. It has the architecture of the classic autoencoder consisting of two neural networks (encoder and decoder) with the same number of layers and neurons in each layer connected through the bottleneck layer with two neurons. In addition to the autoencoder loss $L_a(X, \tilde{X})$, encodermap loss has a term with the sketch-map loss function $L_s(X, x)$, which improves the quality of the two-dimensional projection obtained in the bottleneck layer [see Eq. (2)].

Downloaded from http://pubs.aip.org/aip/jcp/article-pdf/doi/10.1063/5.0142797/16814539/144109_1_5.0142797.pdf

© Author(s) 2023

C. Hierarchical density-based spatial clustering of applications with noise (HDBSCAN)

The HDBSCAN[34,39] can be approached from two different sides: it can be described as a hierarchical implementation of a new formulation of the original DBSCAN[25] algorithm called DBSCAN* by Campello *et al.*,[34] or it can be formulated as a robust version of single-linkage clustering with a sophisticated method to obtain a flat clustering result, as done by McInnes *et al.*[39] Here, we describe it through the second approach.

In the first step, the algorithm introduces the so-called mutual reachability distance (MRD) [Eq. (6)], which transforms the space to make sparse points even sparser but does not significantly change the distance between already dense points,

$$D_{mreach-k}(x_i, x_j) = \max\{core_k(x_i), core_k(x_j), D(x_i, x_j)\}, \qquad (6)$$

where x are points being clustered, k is a constant that determines the number of nearest neighboring points, $core_k(x)$ is the function that finds the maximum distance between a point x and its k nearest neighbors, and $D(\cdot,\cdot)$ is the distance between two points. The maximum of three distances is selected as the MRD [Fig. 3(i)].

In the next step, the minimum spanning tree based on the MRDs is built via Prim's algorithm[40] [see Fig. 3(ii)]. This is done by starting with the lowest MRD in the dataset and connecting the two points with a straight line. In the following steps, always, the next nearest data point to the existing tree, which is not yet connected, is added to the tree.

Once the minimum spanning tree is generated, the cluster hierarchy can be built. This is done by first sorting the edges of the tree by distance. Then, the algorithm iterates over the edges, always merging the clusters with the smallest MRD. The result of this procedure is shown in Fig. 3(iii).

In order to extract a flat clustering from this hierarchy, a final step is needed. In this step, the cluster hierarchy is condensed down, by defining a minimum cluster size and checking at each splitting point if the newly forming cluster has at least the same number of members as the minimum cluster size. If that is the case, then a new cluster is accepted; if not, the data points splitting off are considered noise. The condensed tree of a toy system is shown in Fig. 3(iv).

D. Introduction of a new clustering workflow

In this article, we present a data processing routine that we found to be extremely efficient for large molecular dynamics simulation trajectories. It relies on the three algorithms introduced above. A schematic description is shown in Fig. 1. In this workflow, a given dataset is clustered iteratively until either a specified amount of data points are assigned to clusters or a maximum number of iterations have been reached.

Figure 1 illustrates the sequence of data processing steps along the clustering workflow. In the first step, a high-dimensional collective variable (CV) is chosen. For all systems that are shown in this article, all pairwise distances between the C_α atoms were selected. After a CV has been chosen, for trajectories containing more than

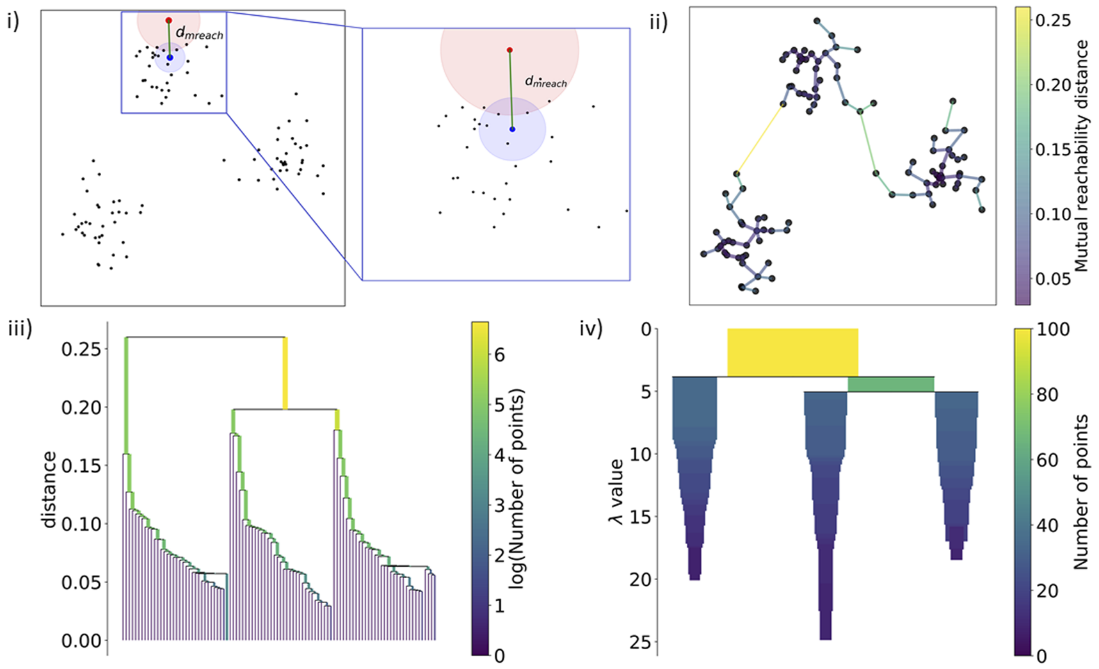


FIG. 3. Application of HDBSCAN to a toy dataset with three clusters. (i) Example for the computation of the MRD for two points (red and blue). The red and blue circles indicate the farthest distance to the five nearest neighbors for both points. One can see that the distance between the red and blue points (green line) is larger than both the radii of the blue and the red circle. Therefore, in this case, the green line distance is chosen as the MRD. (ii) Minimum spanning tree based on the MRDs. (iii) Cluster hierarchy. (iv) Condensed clustering.

Downloaded from http://pubs.aip.org/aip/jcp/article-pdf/doi/10.1063/5.0142797/16814539/144109_1_5.0142797.pdf

© Author(s) 2023

25 000 frames, encodermap is trained on all data. Therefore, we obtain a function that can be used to very efficiently project data into the newly generated 2D space. In parallel, a random subset from the entire dataset is generated. The reason to use such a subset is a limitation that comes with the cc_analysis dimensionality reduction. As mentioned in Sec. II A, the cc_analysis algorithm works with the correlation matrix. This means that the Pearson correlation coefficients of the selected CV (here the pairwise c-alpha distances) are calculated for all unique pairs of frames and used as input to cc_analysis. However, the larger a dataset is, the larger the correlation coefficient matrix will be, until it is no longer efficient to work with that matrix due to very long computation times and memory issues. Therefore, a subset is created, by randomly selecting up to 25 000 data points from the entire dataset. This subset is then used in the cc_analysis dimensionality reduction to project the high-dimensional CVs (between 190 and 1081 dimensions for the systems in this article) to a lower-dimensional subspace (20–30 dimensions for the systems in this article). The choice of the appropriate amount of reduced dimensions is made by searching for a spectral gap among the cc_analysis eigenvalues. Once the cc_analysis space has been identified, a clustering is generated by applying the HDBSCAN algorithm to that lower dimensional data. Even though we have found that the HDBSCAN algorithm works very well in the cc_analysis space, it is, of course, possible to exchange this algorithm with any other density based clustering method (e.g., density peak-based clustering[26]). This again highlights the modularity of the presented workflow. A detailed description of how to choose the dimensionality for cc_analysis and the parameters for HDBSCAN is given in the supplementary material, Sec. S-I.

We use two different DR algorithms in the workflow due to the following reasons. For once, the cc_analysis algorithm is used to project the smaller subsets to a still comparably high-dimensional subspace, which holds more information compared to the 2D projection of encodermap. This higher-dimensional subspace is, therefore, very well suited to be the clustering space. Once the data subset is clustered in the cc_analysis space, the 2D encodermap space is used to assign the points that were not a part of the subset to the found clusters. The 2D projection is very well suited to do a fast assignment of additional points from the dataset and to serve for visualization purposes. Additionally, encodermap is able to project huge datasets very time-efficiently. Hence, the generated 2D projection of a given dataset can be used to avoid the main disadvantage of the cc_analysis algorithm in the way we use the algorithm here, which is having to use subsets of the data due to memory issues. In order to circumvent this disadvantage, we build a convex hull in the 2D space for each cluster that was found in the cc_analysis space. If an unassigned point lies inside a convex hull, the RMSD to the central conformation of that cluster is computed. This convex hull criterion, therefore, works as an acceleration element in the algorithm, since it ensures that the RMSD does not need to be computed for every single data point in the remaining dataset. The acceleration obtained via this approach is discussed in Sec. IV. In case the RMSD is inside a given cutoff, the data point is considered to be part of that cluster; otherwise, it is not assigned to the cluster. This RMSD cutoff is chosen by taking the weighted mean of all average internal cluster RMSDs[41] of the first clustering iteration. We found that this procedure generates structurally quite well defined clusters with a low internal cluster RMSD since the RMSD criterion is based on well defined conformational states that emerged from cc_analysis combined with HDBSCAN. However, there is also the possibility to identify more fuzzy clusters that only share a general structural motif by using a larger RMSD cutoff for the assignment. An example of the identification of such fuzzy clusters is described in Sec. III B.

By introducing a RMSD criterion in the last step, we force the clustering to only include structurally very similar conformations in the respective clusters. There are, of course, various other clustering algorithms that group MD datasets based on their RMSD values, e.g., the implementation[28] in the GROMACS software package.[42] Such RMSD-based clustering algorithms have been used in the MD community for at least 20 years now, and they are a very obvious choice for conformational clusterings of MD trajectories. They directly compare the positions of specified atoms in various conformations of a molecule and then group the individual conformations along the trajectory using a cutoff value. However, these methods generally rely on the full RMSD matrix of a given dataset. For larger trajectories, it becomes almost infeasible to compute these matrices due to extremely long computation times and memory issues that arise when working with such large matrices. In our workflow, we can circumvent these issues by only having to compute the RMSD between the coordinates of C_α atoms of the central conformations of each cluster and the data points that lie inside the convex hull of the respective clusters in the 2D space.

In case a given system has less than about 50 000 frames, it is, in principle, also possible to omit the encodermap part, since the cc_analysis algorithm is able to handle the entire dataset. If this approach is chosen, the user can either entirely skip the RMSD criterion or the members of clusters that are found in the cc_analysis space can still be accepted/rejected based on a RMSD cutoff. An advantage of using both the cc_analysis algorithm and the encodermap algorithm together is the possibility to check the dimensionality reduction steps on the fly. Since the clustering is done in one DR space but visualized in the other, narrow and well defined clusters in the 2D space indicate that the 2D map separates the different conformational clusters nicely and that, therefore, the chosen encodermap parameters were well selected.

Our clustering scheme is not very dependent on the quality of the encodermap projection, as it is used only to assign additional structures to already identified clusters. Since the clustering itself is done in the higher-dimensional cc_analysis space, the final cluster assignment uses a RMSD cutoff. The only requirement that the scheme poses toward the 2D map is that similar conformations are located close to each other on the map. This is achieved by the MDS-like distance loss part of the overall loss function of encodermap.

The remaining points that were not assigned to any cluster after one clustering iteration are then used as a new pool of data from which the new random subset is built. This whole cycle is repeated until a certain amount of data points are assigned to clusters or until a certain number of clustering iterations are performed. To decide on a stopping point for the iterative procedure, we rely on two possible convergence criteria: either a percentage of assigned conformations or average cluster sizes found at an iteration. If we observe a plateau in the percentage of unassigned data points during several successive iterations, the clustering procedure is stopped. Due to the design of our workflow, the average cluster size of newly added

Downloaded from http://pubs.aip.org/aip/jcp/article-pdf/doi/10.1063/5.0142797/16814539/144109_1_5.0142797.pdf

© Author(s) 2023

clusters will decrease with each iteration. Therefore, the average size of newly added clusters or the convergence of this property during successive iterations can also be used as a stopping criterion. Examples are shown in the supplementary material, Sec. S-II, Fig. S2.

The conformational clusters that are identified by this workflow can be used in various different ways, for example, to build a kinetic model, which is discussed more in Sec. IV. Another use case would be to take advantage of the knowledge gained from the clustering to initiate and/or evaluate enhanced sampling schemes[43,44] to sample lowly populated parts of phase space in order to speed up the overall convergence, or the found clusters might simply be used to get an overview of the conformational variety that is present in any given dataset, which works especially well in combination with the visualization of the accessible phase space in the 2D encodermap projection.

III. RESULTS

A. Description of the proteins' trajectories used for the analysis

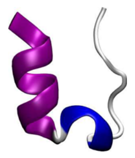
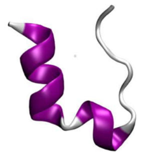
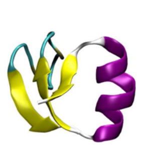
In order to illustrate the capability and performance of the proposed scheme, we chose four test systems: one is the 40 temperature replica exchange (RE) trajectories of the Trp-cage protein (TC5b) analyzed in the original encodermap paper;[33] the other three systems are long trajectories of Trp-cage (TC10b), NTL9, and Protein B simulated by the Shaw group on the Anton supercomputer[45] and generously provided by them. The four systems are listed in Table I. For all the systems, we chose distances between C_α atoms as the input collective variables.

The first protein we analyze in this work is the Trp-cage system (TC5b) (Trp-cage RE). It is a comparatively small protein (20 residues), which has a very stable native state when simulated at room temperature. The combination of 40 temperature replica exchange trajectories (temperature range from 300 to 570 K, 3.2 μs of simulation time, 1577 520 frames) gives a very diverse mixture of structures, including trajectories where the system is very stable and barely moves away from the native state, as well as highly disordered trajectories where high-energy conformations are visited. This combination of conformations makes the data set extremely diverse and complicated for analysis due to the high number of expected clusters with extremely varying sizes and densities.

Second, we consider the K8A mutant of the thermostable Trp-cage variant TC10b (Trp-cage Anton) simulated by Lindorff–Larsen *et al.*[45] (208 μs; 1044 000 frames). This simulation was run at 290 K and produced a much more disordered trajectory compared to the low temperature replica simulations of the TC5b system. Despite the fact that TC5b and the K8A mutant of TC10b have slightly different amino acid sequences, we use the same trained encodermap to project both systems in the same 2D map (see Figs. 4 and 5), since both systems have the same number of residues and, therefore, the same dimensionality of CVs. This offers the opportunity to demonstrate that different systems can be compared to each other very nicely when projected into the same 2D space.

Next, we probed our clustering scheme with extremely long (1877 μs;[46] 9389 654 frames) simulations[45] of the larger (39 amino acids) N-terminal fragment of ribosomal protein L9 (NTL9), which has an incredibly stable native state. Besides the possibility to show how the algorithm deals with this extremely large dataset, the system has also been studied by several other researchers.[29,47] This allows us to compare our results to their findings. Schwantes and Pande[47] reported on very low populated states that contain register-shifts between the residues that are involved in the formation of the beta sheet structures of NTL9. This opens up the opportunity to show whether our clustering workflow is able to identify both very large states and extremely lowly populated states in the same dataset.

TABLE I. Proteins analyzed in this study and performance overview of the clustering scheme.

	Trp-cage RE (TC5b)	Trp-cage Anton (TC10b)	NTL9	Protein B
				
Trajectory length in μs	3.2	208	1877	104
Number of frames	1 577 520	1 044 000	9 389 654	520 250
Dimensionality of input CVs	190	190	703	1081
Number of cc_analysis dimensions	20	20	20	30
Average iteration time on our local workstation (see the supplementary material, Sec. S-V) (min)	15	18	55	12
Average iteration time over all used CPU threads (min)	24 × 15 = 360	24 × 18 = 432	4 × 55 = 1320	24 × 12 = 288
Frames assigned to clusters after ten iterations	60.5%	33.1%	80.9%	20%
Total CPU time over all iterations (min)	3600	4320	13 200	2880

Downloaded from http://pubs.aip.org/aip/jcp/article-pdf/doi/10.1063/5.0142797/16814539/144109_1_5.0142797.pdf

© Author(s) 2023

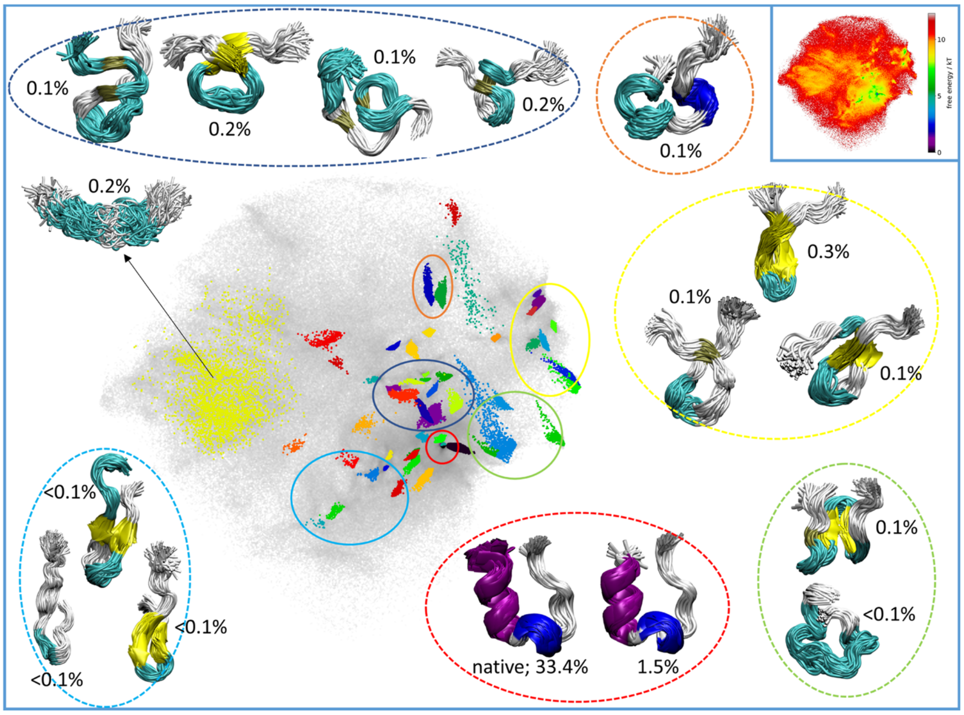


FIG. 4. Trp-Cage TC5b (40 temperature RE trajectories): exemplary conformations of the most populated clusters found in each of the areas indicated by colored circles and their populations in percentages. The cluster representatives show the average secondary structure over the entire cluster. The clusters are colored randomly, and the colors repeat. Therefore, clusters that have the same color but are separated in 2D space contain different conformations. The depicted clusters hold 36.5% of all conformations. Most of the remaining 24% of conformations that have been assigned to clusters are slight variations of the native structure and are not shown here due to visibility reasons. The cluster that is referred to by an arrow is one of the fuzzy clusters that were generated by increasing the RMSD cutoff. Top right: a histogram of the 2D encodermap space.

Finally, we chose to analyze the protein B simulations (104 μs; 520 250 frames).[45] Compared to the aforementioned proteins, protein B does not have a single very stable state; instead, three helices can move quite easily against each other. This leads to a broad conformational space where the energy barriers between the individual states are very small. Therefore, the individual conformational states are not as easily separable and rather fade/transition into each other. Taking the long simulation time into account, this system is very hard to cluster conformationally.

To demonstrate how our clustering scheme works, we chose to apply it to these four systems that pose very diverse challenges (e.g., an extremely large dataset, both highly and very lowly populated states in the same data, and differences in the amount of folded/unfolded conformations along the trajectories). For each of the systems, we initially conducted the same number of clustering iterations (10), then evaluated the resulting clustering, and decided whether, for a given system, additional iterations were needed.

B. Trp-cage

1. TC5b

For the RE simulations of the Trp-cage, the clustering scheme was run over ten iterations and 60.5% of all conformations were assigned to clusters. Figure 4 shows an encodermap projection of all 40 replicas with some of the most populated clusters found after ten iterations and representative conformations of these clusters. Similar conformations are grouped together, and rare structures are spread out across the map. For example, the native conformation of Trp-cage RE (33.4% of all conformations) is shown in the bottom right of the 2D map in Fig. 4. On the bottom left, conformations with one turn near the middle of the backbone are located. The two parts of the backbone chain of these conformations lie right next to each other and partially form beta-sheet structures.

Using a larger cutoff distance in the RMSD-based assignment of structures to the clusters (the other clusters were generated by applying a 1.8 Å RMSD cutoff to the central conformation), we obtained larger and quite diffuse clusters of extended conformations (one of these clusters is shown in the left part of the projection in Fig. 4, where it is referred to by an arrow). An appropriate size of this RMSD cutoff was defined for each fuzzy cluster individually by computing the mean value of the largest 20% of the RMSD values between the centroid and cluster members of the cluster identified in the current iteration (it is equal to 5.5 Å for the cluster shown here). Before we identify fuzzy clusters, we first continuously assign structures based on a fixed RMSD cutoff (1.8 Å for TC5b) until one of the stopping points defined in Sec. II D is reached (average cluster size for TC5b). Once this stopping point is reached, the RMSD cutoff is adjusted in the way explained above and fuzzy clusters are obtained. Thereby, one ensures that all conformations that can be assigned to well-defined clusters are removed from consideration before identifying fuzzy clusters. The usage of such a varying cutoff can be very helpful in order to identify diffuse clusters, where the members share a certain structural motif but do not converge to a very defined conformation, just like the cluster shown here.

From the clustering results shown in Fig. 4, one can see that the proposed clustering workflow manages to efficiently identify structurally very well defined clusters for the TC5b system. Over ten clustering iterations, it assigned 60.5% of all conformations to

Downloaded from http://pubs.aip.org/aip/jcp/article-pdf/doi/10.1063/5.0142797/16814539/144109_1_5.0142797.pdf

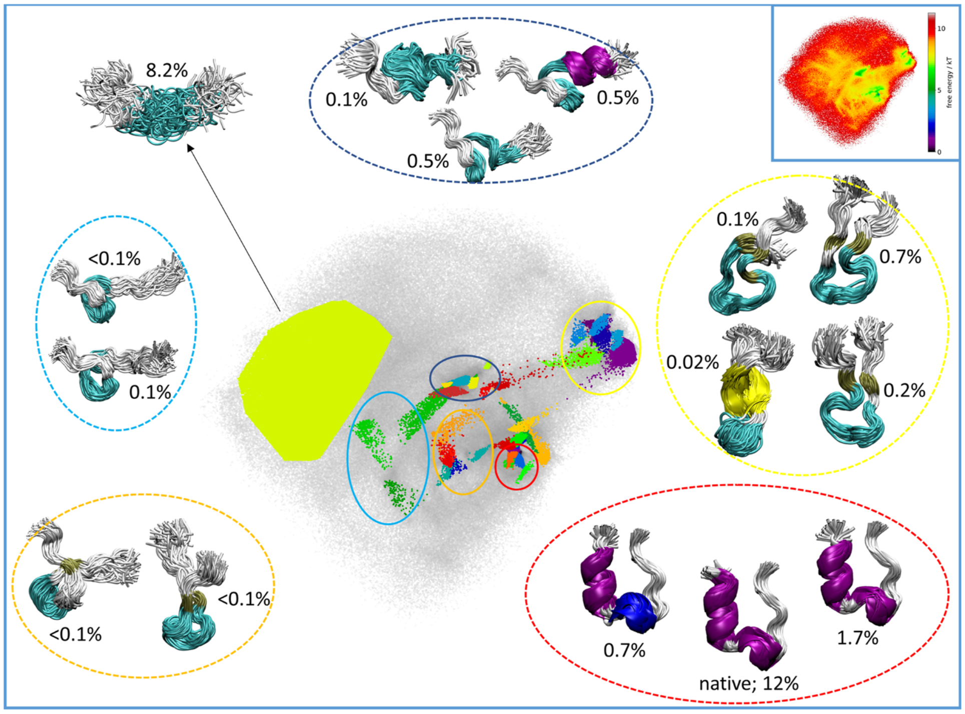


FIG. 5. Most populated clusters and respective conformations of Trp-Cage TC10b[45] projected to the same 2D encodermap space as TC5b (Fig. 4). Top right: a histogram of the 2D projection.

260 clusters. Besides the highly populated native state (33.4%), the algorithm also finds very "rare" states, which contain only a very small amount of conformations ($\leq 0.1\%$) but, nevertheless, show a very defined structural identity.

2. TC10b

Figure 5 shows the same analysis applied to the trajectory of the K8A mutant of TC10b Trp-cage. We used the encodermap algorithm that we trained on TC5b to project the trajectories to the same 2D space. The identification of clusters, however, is entirely independent and unique for both cases, since the clustering is done in the higher-dimensional cc_analysis space.

Notably, the backbone conformation of the native state of this mutant is extremely similar to the one in the TC5b system. However, this biggest cluster only contains 12% of all conformations along the trajectory, compared to the 33.4% in the case of the TC5b system. If all clusters whose central conformation is within 2 Å RMSD to the native conformation are combined, we get a native conformation percentage of 16.9%. This is in excellent agreement with the native cluster sizes reported by Deng *et al.*[48] and Ghorbani *et al.*[49] who analyzed the same Trp-cage trajectories provided by Lindorff–Larsen *et al.*[45] Furthermore, our 33.4% of assigned conformations coincide very well with the reporting of Sidky *et al.*[50] They found a total of 31% of conformations distributed over eight metastable macrostates and the remaining 69% as one big "molten globule" state.

The TC10b trajectory is more disordered; this can be seen by the more homogeneous projection in 2D space (the upper right plot in Fig. 5) and the RMSD values to the native conformation in the supplementary material, Sec. S-III, Fig. S3. This is also the reason why the clustering scheme assigned only 33.4% of all conformations to clusters after ten iterations. If more frames should be assigned to clusters, more clustering iterations can be performed, the RMSD cutoff can be increased, or both can be done simultaneously (for the Protein B system, we show the results of this approach later in the article).

However, the clusters in the very center of the map (the dark blue circle) are much more compact and collapsed compared to the clusters that were found in the similar area of Trp-cage RE's 2D projection. In addition, some of the clusters that were found at the very bottom of the left-hand side of the map in the case of the replica trajectories (the light blue circle) were not found at all in the TC10b trajectory. The very large and diffuse cluster on the left-hand side of the map is present in both systems as well.

3. Clustering directly in 2D space of TC5b

The clustering discussed above was performed in a 20-dimensional space after applying the cc_analysis algorithm and was only displayed in a 2D projection done with encodermap. In order to demonstrate the advantages of our approach, we also directly clustered the 2D encodermap space using HDBSCAN. The encodermap space that we used for this clustering is the same space that we used to visualize the cc_analysis clustering in Figs. 4 and 5. The results of this clustering and a few chosen clusters are shown in Fig. 6. In total, this clustering assigned 13.5% of all conformations to 362 clusters. The biggest cluster that was found is the native cluster; however, it only contains 0.8% of all conformations compared to the 33.4% that were found by clustering the cc_analysis space. The clustering in the 2D space identifies some structurally very well defined clusters, such as clusters 0, 1, and 3, but also a lot

© Author(s) 2023

Downloaded from http://pubs.aip.org/aip/jcp/article-pdf/doi/10.1063/5.0142797/16814539/144109_1_5.0142797.pdf

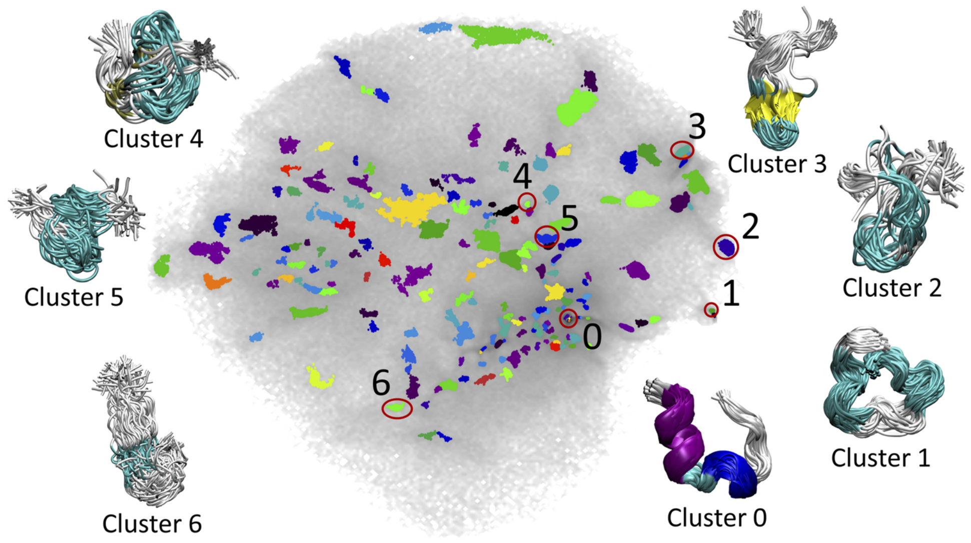


FIG. 6. 2D encodermap space of TC5b clustered with HDBSCAN. Representations of chosen clusters that have the same location in the 2D map as clusters found with the clustering scheme in Fig. 4 are shown.

of very diffuse and inhomogeneous clusters. To quantify this inhomogeneity, we computed the average of the internal cluster RMSDs. For the TC5b system, our clustering workflow resulted in an average cluster RMSD of 1.34 Å and a weighted average RMSD of 1.03 Å, where weights are defined as the fraction of each cluster to all clustered data. The average RMSD for the direct clustering in the 2D space is 2.25 Å, and the weighted average RMSD is 2.73 Å. This clearly shows that the internal cluster RMSD variance is, on average, much larger when clustering directly in the 2D space. Furthermore, the clustering in the 2D space itself naturally highly depends on the quality of the 2D map.

Other than the much clearer conformational identity of the individual clusters (shown via internal cluster RMSDs), our clustering scheme also manages to assign 60.5% of all conformations to different clusters. Compared to that, the clustering in the 2D projection only assigned 9%–14% of all conformations, depending on the choice of clustering parameters.

4. Comparison to other clustering approaches

For a further assessment of our clustering scheme, we have also applied a frequently used clustering routine to the TC5b data. In the supplementary material, Sec. S-IV and Figs. S4 and S5, the results of applying the k-means algorithm to an 11-dimensional PCA projection of the same CVs (pairwise C_α distances of TC5b) are shown.

In summary, the scheme identified both structurally very defined and quite diffuse clusters in considered systems. Even though the combination of the 40 RE trajectories produces a very diverse dataset, the clustering scheme manages to assign a large amount of the conformations to clusters (60%). Our clustering results for the TC10b are in very good agreement with the findings of other researchers.[48–50] Furthermore, the comparison to a clustering in the 2D space clearly shows the superiority of using more dimensions obtained with the cc_analysis algorithm in HDBSCAN over just relying on a low-dimensional representation alone.

C. NTL9

Next, we examined very long (1877 μs) simulations of NTL9.[45] With 9.38×10^6 frames to cluster, this system is an ideal candidate to demonstrate how the proposed algorithm copes with large amounts of data. After ten iterations, 81% of all conformations were assigned to clusters. Figure 7 shows a 2D projection made with encodermap, where points are colored according to the clusters found after ten iterations of the scheme, and a histogram of the 2D space is in the upper right corner. In total, we found 157 clusters and assigned to them 81% of all conformations over ten clustering iterations.

A comparison of the time series of the RMSD values for the folded state with the respective data of the Trp-cage Anton simulations (the supplementary material, Sec. S-III, Fig. S3) reveals that the two systems exhibit very different dynamics. While in the Trp-cage case, the RMSDs show the disordered nature of the system, in the case of the NTL9 trajectories, the RMSDs are predominantly quite low and only spike up to larger values for rather short time periods. This suggests that the NTL9 system resides in a native-like state for the majority of the simulated time. This was confirmed during the very first iteration of the clustering scheme. There we found two clusters that make up 75.8% of all conformations.

This example also nicely illustrates how the iterative clustering approach can be efficient in identifying clusters of very different sizes and densities (highly populated native states and low populated clusters). After finding and removing the first two clusters (75.8% of the data), the clustering algorithm becomes much more sensitive toward the less dense areas in the CV-space in the following clustering iterations.

We compared our clustering results with those other publications analyzing the NTL9 trajectories from Ref. 45. Mardt *et al.*[29] applied VAMPnets to trajectory 0 and found, in total, 89.1% of folded, native-like conformations. If we take the clusters we found by analyzing the trajectories 0, 2, and 3 and evaluate the conformations stemming from trajectory 0 (trajectory 0 resides in the native-like state for a larger fraction of the simulated time; see RMSD plots in the supplementary material, Sec. S-III, Fig. S3), the

© Author(s) 2023

Downloaded from http://pubs.aip.org/aip/jcp/article-pdf/doi/10.1063/5.0142797/16814539/144109_1_5.0142797.pdf

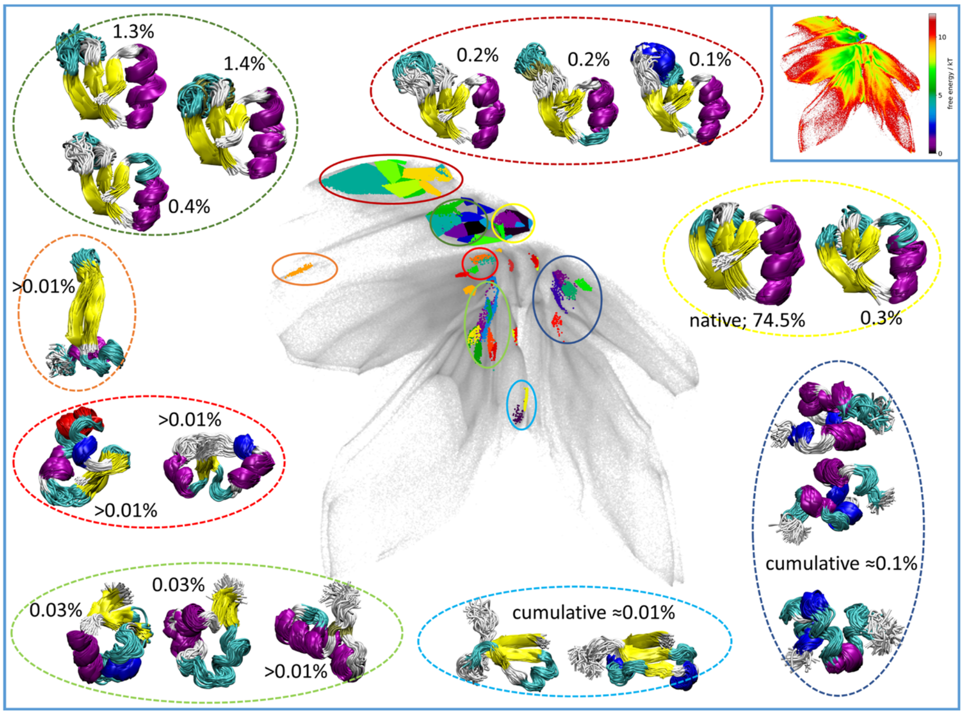


FIG. 7. 2D encodermap projection of NTL9. The projection can be approximately divided into three parts: the upper part with the densest areas (where the native-like states are located) and the lower left and right planes divided by an unpopulated vertical gap. The left-hand side includes various conformations with a singular beta-sheet formed mostly between the beginning and the end of the protein. In contrast, on the right-hand side lie the mostly extended conformations with multiple helices along the backbone. Exemplary conformations of some of the most populated clusters found in each of the marked areas on the map and their populations are shown. All clusters in the yellow circle are extremely similar to the native cluster and can be summed up to a total of 76% of all conformations. The structures that are shown here make up 78.4% of all conformations. Top right: histogram of the 2D encodermap space.

amount of folded, native-like conformations we find is in very good agreement with that of Mardt *et al.*[29] Furthermore, Schwantes and Pande[47] reported the finding of three "register-shifted" states, which are very low populated and, therefore, very hard to find. "Register-shifted" refers to the identity of the specific residues involved in forming the beta-sheet structure in the native-like states (residues 1–6, 16–21, and 35–39). With our method, we identified six different register-shifted states in the NTL9 trajectories 0, 2, and 3 (see Fig. 8).

States 0, 1, and 2 are the ones that were also found in Ref. 47. To our knowledge, states 3, 4, and 5 have not been reported yet. In state 0, the central of the three beta-sheet strands is shifted downward, whereas in state 2, the rightmost strand is shifted downward. In state 1, both the middle and the rightmost strands are dislocated compared to the native state. State 3 is similar to state 1 in the fact that both the middle and the rightmost strands are shifted; however, in state 3, the rightmost strand is shifted upward and not downward like in state 1. Among these six states, state 4 is unique since the

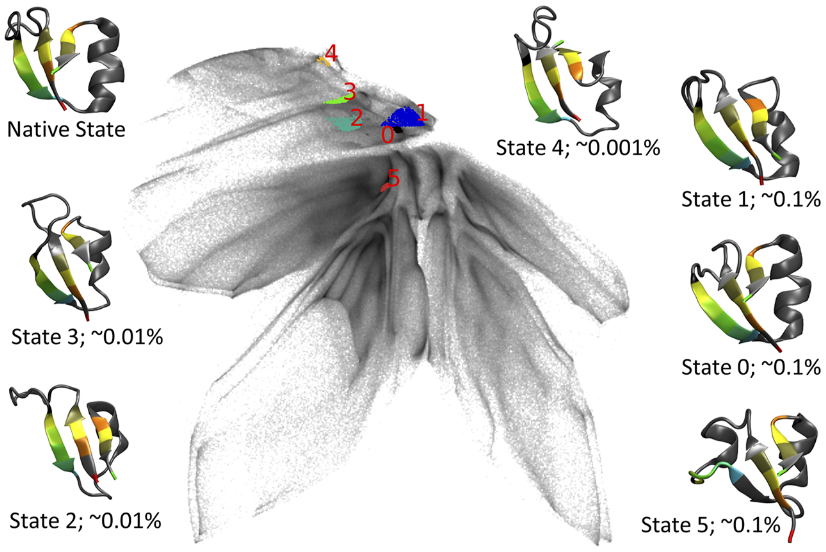


FIG. 8. Register-shifted states found in the NTL9 trajectories 0, 2, and 3. The residues that form the beta-sheets in the native state are colored based on their residue ID.

Downloaded from http://pubs.aip.org/aip/jcp/article-pdf/doi/10.1063/5.0142797/16814539/144109_1_5.0142797.pdf

rightmost strand is turned by 180°. Finally, state 5 differs from other states in having an extra helix along the chain between the leftmost and the middle strand. Because of this additional helix, the leftmost strand is extremely shifted compared to its native state.

The identification of these register-shifted states highlights one asset of the proposed workflow. It is able to find both very large states (native, 74.5%) and very low populated clusters (<0.001%) in the same dataset.

D. Protein-B

The last system we analyzed is protein B. This system does not have a very stable native state; instead, the three helices can move against each other relatively freely. This can be seen in the time series of the RMSD to the closest experimental homologue (1PRB) shown in the supplementary material, Sec. S-III, Fig. S3. There are no extended periods where the values are stable over some time, meaning that there are no large free-energy barriers separating the various accessible conformations and, thus, the system constantly transitions into different conformations. This has also been found in Ref. 45, where the authors stated that they were unable to identify a free-energy barrier between folded and unfolded states for protein B (tested over many different reaction coordinates).

Such a highly dynamic system is very challenging for conformational clustering. Here, we want to show where our algorithm has its limitations and what can be done to get a satisfactory clustering result. Figure 9 gives an overview of some of the clusters found after ten iterations of the scheme. These clusters include only 20% of the protein B trajectory and, thus, 80% of all conformations are still unclustered.

In order to have more data assigned to clusters, two parameters can be adjusted. First, the RMSD cutoff value can be increased and, thereby, more conformations can be assigned to the found clusters. In this specific case, this adjustment is justified since, due to the low free-energy barriers between different states, the individual clusters are not as sharply defined in terms of their conformations. In the ten clustering iterations, which are shown in Fig. 9, we used a RMSD cutoff of 3.0 Å. In a second run, we increased it to 3.5 Å. This resulted in an assignment of 31% of all conformations to generally more loosely defined clusters.

A second approach is to increase the number of clustering iterations. For the first ten clustering iterations of previously analyzed systems, we manually tuned the clustering parameters. This includes the choice of the number of cc_analysis dimensions as well as the min_samples and min_cluster_size parameters of HDBSCAN. However, such a manual adjustment of the parameters is, of course, not feasible for automating the script in order to perform many more clustering iterations. Since the amount of cc_analysis dimensions needs to be very rarely changed once a suitable amount has been identified in the first clustering iteration, the automation of the script only relies on the choice of the HDBSCAN parameters. Once the number of clusters found in a single iteration falls below a certain threshold (ten clusters in this case), the numerical values of the min_samples and min_cluster_size parameters of HDBSCAN are slightly decreased. This leads to the detection of smaller clusters that have not been identified before. By applying this automation approach after the first ten iterations to protein B and using a RMSD cutoff of 3.5 Å, we could assign 44.3% of all conformations to clusters over 100 iterations, which took roughly 15 h on our workstation.

IV. DISCUSSION

The Trp-cage system (TC5b) is a relatively small protein that has a quite stable native conformation. The combination of 40

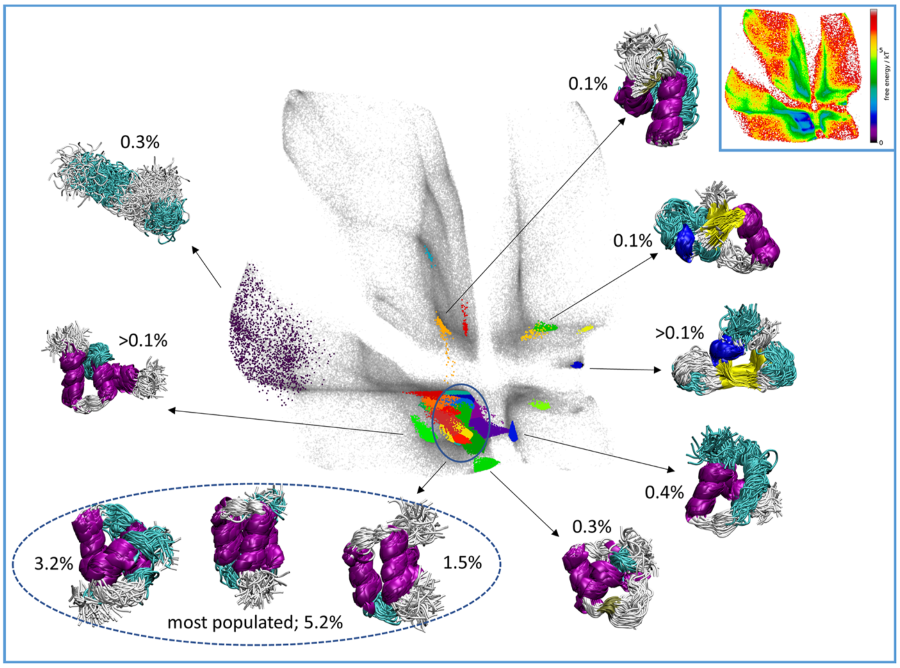


FIG. 9. Protein B: exemplary conformations of some of the most populated clusters found for the protein B system after ten clustering iterations and their populations. Top right: histogram of the 2D encodermap space.

Downloaded from http://pubs.aip.org/aip/jcp/article-pdf/doi/10.1063/5.0142797/16814539/144109_1_5.0142797.pdf

© Author(s) 2023

temperature RE trajectories, however, gives a very diverse dataset including (under standard conditions) very improbable high-energy conformations. Over ten iterations, the algorithm managed to assign 60.5% of all conformations to clusters, which took, on average, 360 min per iteration over all CPU threads (15 min per iteration on a standard office machine with 24 CPU threads). Table I shows the clustering performance for the four systems discussed here. By switching the generally static RMSD cutoff to a varying cutoff, we could show that the algorithm can generate both conformationally very defined clusters and clusters that are quite diffuse. The conformations assigned to such loose clusters share a general structural motif. The ability to identify both of these cluster types is one of the advantages of the proposed algorithm. Furthermore, we demonstrate that the clustering workflow is able to directly compare different systems (even if they slightly differ structurally), by projecting them to the same 2D map using the encodermap algorithm. This enables a direct and visual comparison of the sampled phase-spaces of different trajectories and their respective identified states. By comparing the clustering result where the clustering is done in a 20-dimensional cc_analysis space and then projected to a two-dimensional space to a clustering where the clusters are purely found in a 2D encodermap space, we prove an advantage using more dimensions and combine cc_analysis with encodermap. The scheme created clusters with a much clearer structural identity (a lower RMSD variance) while being much less dependent on the quality of the 2D map.

We analyzed long (9.38×10^6 frame) trajectories of NTL9 to show how the proposed scheme copes with very large amounts of data. On average, the algorithm needed 1320 min of computation time over all CPU threads per iteration (55 min per iteration on our office machine). Since this system also has one hugely populated native state, it is also a nice example to demonstrate an advantage of iterative clustering. After the clusters with the native states are removed from consideration, the algorithm becomes much more sensitive toward less populated areas in the following iterations. Applying this approach, we could identify three very low populated register-shifted states, which have been reported before,[47] and three not yet seen register-shifted states.

Finally, we looked at protein B, which is a highly dynamic system. To analyze this 1.04×10^6 frame trajectory, it took, on average, 288 min of computation time per iteration (12 min per iteration on our office machine). This system has no large free-energy barriers separating the various conformations, which makes it very difficult to cluster. This was confirmed by the fact that after ten clustering iterations, only 20% of all conformations could be assigned to clusters. However, by increasing the RMSD cutoff from 3.0 to 3.5 Å, we could already increase the number of assigned conformations to 31%, which, of course, resulted in slightly less structurally defined clusters. It is also possible to automate the clustering and run until a certain number of conformations are assigned to clusters or until the given number of iterations is reached. In this specific case, we ran the scheme for 100 automated iterations (≈15 h), during which 44.3% of the conformations were assigned to clusters.

For all considered systems, the proposed workflow was able to identify defined clusters at the cost of leaving some amount of the trajectories unassigned. As we have shown here, the rest of the structures do not belong to any specific clusters and can be considered as unfolded or transition states. We intentionally do not propose any additional steps to assign or classify those conformations as it is highly dependent on the intended application of the data. For example, in case the data are used to build subsequent kinetic models, the rest of the points can be assigned to the nearest (e.g., in simulation time) cluster using methods such as PCCA + analysis,[51] or defined as a metastable transition state as in Ref. 50. It can also be defined as noise and used as discussed in Ref. 52. If such a kinetic model were to be built from the conformational clusters that we identify, e.g., for the TC5b system, most likely, many of the smaller clusters found in the disordered part of phase space would be combined to form just a very few macrostates.

All performance data are shown in Table I and were obtained by running the clustering scheme script on the office workstation described in the supplementary material, Sec. S-V. The proposed workflow is, however, highly parallelizable, since the computationally most expensive step is the assignment of additional data points to the initially identified clusters in the small subset based on the convex hull and the RMSD criterion. If a large number of CPU cores are available, the 2D encodermap projection array can be split by the number of cores and the assignment can, thereby, be run in parallel, leading to a significant speedup.

The convex hull around the clusters identified in the small subset is used to reduce the amount of RMSD computations that have to be performed when assigning additional conformations in each clustering iteration. This, however, might, in principle, lead to the exclusion of data points that might otherwise have been assigned to some of the clusters. In order to get an idea of the magnitude of this "loss" of potential cluster members, we computed the RMSD of all data that were labeled as noise (623 000 conformations; 39.5%) to each of the cluster centers of TC5b (260 clusters). This computationally very expensive task took an additional 5 h on our working machine. We found that 42 000 conformations (2.7%) were not assigned to the identified clusters due to the convex hull criterion. When keeping in mind that the entire ten iteration clustering process took 2.5 h, the "loss" of 2.7% of unclustered data can be considered a worthy trade-off.

Another point to consider is that due to the convex hull criterion, clusters can be split. If data points that would be assigned to a certain cluster by reason of the RMSD criterion lie outside of the convex hull, they could be identified as another cluster in one of the following clustering iterations. In such cases, it can make sense to merge these clusters in hindsight due to their very similar structural identity. In order to showcase such a merge, we again analyzed TC5b. We computed the RMSDs between all of the 260 central cluster conformations and merged all clusters that had a RMSD of ≤1 Å. This resulted in a reduction to 201 clusters with only a very marginal influence on the average internal cluster RMSDs.

The code for the encodermap algorithm is available on the following GitHub page: https://github.com/AG-Peter/encodermap. The cc_analysis code can be found at https://strucbio.biologie.uni-konstanz.de/xdswiki/index.php/Cc_analysis.

V. CONCLUSION

We developed a clustering scheme that combines two different dimensionality reduction algorithms (cc_analysis and encodermap) and HDBSCAN in an iterative approach to perform fast and accurate clustering of molecular dynamics simulations' trajectories. The

Downloaded from http://pubs.aip.org/aip/jcp/article-pdf/doi/10.1063/5.0142797/16814539/144109_1_5.0142797.pdf

© Author(s) 2023

cc_analysis dimensionality reduction method was first applied to protein simulation data. The method projects collective variables to a usually relatively high-dimensional (~10–40 dim) unit sphere, separating noise and fluctuations from important structural information. Then, the data can be efficiently clustered by density based clustering methods, such as HDBSCAN. The iterative application of HDBSCAN allows us to account for the inhomogeneity in population and density of the projected points, which is very typical for protein simulation data. As cc_analysis relies on the calculation of correlation matrices between each frame, this drastically limits the amount of data one can project simultaneously. To allow the processing of long simulation trajectories, we included encodermap to the scheme. In addition to the obvious advantage of the two-dimensional visualization, it is used in combination with a RMSD-based acceptance criterion for a fast structure-based assignment of additional points to the clusters initially identified in the higher-dimensional projection done with cc_analysis. To demonstrate the accuracy and performance of the proposed scheme, we applied the clustering scheme to four test systems: replica exchange simulations of Trp-cage and three long trajectories of a Trp-cage mutant, NTL9, and protein B generated on the Anton supercomputer. By applying the scheme to these four test systems, we could show that the algorithm can efficiently handle very large amounts of data, that it can be used to compare the clusters of structurally different systems in one 2D map, and that it can also be applied to cluster systems that do not have very stable native states and are, therefore, intrinsically very difficult to cluster conformationally. Furthermore, the algorithm is able to find clusters independent of their size. By varying the RMSD cutoff, both conformationally very well defined clusters and fuzzy clusters, whose members only share an overall structural motive, can be identified.

SUPPLEMENTARY MATERIAL

The supplementary material (PDF) includes the following:

S-I: Methods to choose parameters for cc_analysis and HDBSCAN.

S-II: Stopping criteria for the clustering workflow.

S-III: RMSD plots of trajectories for Trp-cage, Protein B, and NTL9.

S-IV: Comparison of the proposed clustering workflow to PCA and k-means clustering for Trp-cage (TC5b).

S-V: Workstation specifications.

ACKNOWLEDGMENTS

This work was supported by the DFG through CRC 969. We also greatly appreciate the computing time on bwHPC clusters, which was used to produce the Trp-cage TC5b trajectories. Furthermore, we would like to thank the D. E. Shaw research group for providing the Trp-cage, NTL9, and Protein B trajectories.

AUTHOR DECLARATIONS

Conflict of Interest

The authors have no conflicts to disclose.

Author Contributions

Simon Hunkler: Conceptualization (equal); Formal analysis (lead); Methodology (equal); Software (lead); Visualization (lead); Writing – original draft (lead); Writing – review & editing (equal). **Kay Diederichs**: Methodology (equal); Software (supporting); Writing – review & editing (equal). **Oleksandra Kukharenko**: Conceptualization (equal); Methodology (equal); Writing – original draft (supporting); Writing – review & editing (equal). **Christine Peter**: Conceptualization (equal); Methodology (equal); Resources (lead); Supervision (lead); Writing – review & editing (equal).

DATA AVAILABILITY

Raw simulation data were provided by Tobias Lemke[33] and the D. E. Shaw research group.[45] Derived data supporting the findings of this study are available from the corresponding author upon reasonable request.

REFERENCES

[1] R. O. Dror, C. Young, and D. E. Shaw, "Anton, a special-purpose molecular simulation machine," in *Encyclopedia of Parallel Computing*, edited by D. Padua (Springer US, Boston, MA, 2011), pp. 60–71.

[2] D. E. Shaw, J. P. Grossman, J. A. Bank, B. Batson, J. A. Butts, J. C. Chao, M. M. Deneroff *et al.*, "Anton 2: Raising the bar for performance and programmability in a special-purpose molecular dynamics supercomputer," in *International Conference for High Performance Computing, Networking, Storage and Analysis, SC'14* (IEEE Press, 2014), pp. 41–53.

[3] S. Kmiecik, D. Gront, M. Kolinski, L. Wieteska, A. E. Dawid, and A. Kolinski, "Coarse-grained protein models and their applications," Chem. Rev. **116**, 7898–7936 (2016).

[4] Y. I. Yang, Q. Shao, J. Zhang, L. Yang, and Y. Q. Gao, "Enhanced sampling in molecular dynamics," J. Chem. Phys. **151**, 070902 (2019).

[5] M. Shirts and V. S. Pande, "Screen savers of the world unite!," Science **290**, 1903–1904 (2000).

[6] K. Pearson, "LIII. On lines and planes of closest fit to systems of points in space," Philos. Mag. **2**, 559–572 (1901).

[7] H. Hotelling, "Analysis of a complex of statistical variables into principal components." J. Educ. Psychol. **24**, 417–441 (1933).

[8] L. Molgedey and H. G. Schuster, "Separation of a mixture of independent signals using time delayed correlations," Phys. Rev. Lett. **72**, 3634–3637 (1994).

[9] A. Hyvärinen, J. Karhunen, and E. Oja, "Methods using time structure," in *Independent Component Analysis* (John Wiley & Sons, Inc., 2002), Chap. 18, p. 344.

[10] G. Young and A. S. Householder, "Discussion of a set of points in terms of their mutual distances," Psychometrika **3**, 19–22 (1938).

[11] W. S. Torgerson, "Multidimensional scaling: I. Theory and method," Psychometrika **17**, 401–419 (1952).

[12] M. Ceriotti, G. A. Tribello, and M. Parrinello, "Simplifying the representation of complex free-energy landscapes using sketch-map," Proc. Natl. Acad. Sci. U. S. A. **108**, 13023–13028 (2011).

[13] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," Science **290**(5500), 2319–2323 (2000).

[14] R. R. Coifman, S. Lafon, A. B. Lee, M. Maggioni, B. Nadler, F. Warner, and S. W. Zucker, "Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps," Proc. Natl. Acad. Sci. U. S. A. **102**, 7426–7431 (2005).

[15] R. R. Coifman and S. Lafon, "Diffusion maps," Appl. Comput. Harmon. Anal. **21**, 5–30 (2006).

[16] L. McInnes, J. Healy, N. Saul, and L. Grossberger, "UMAP: Uniform manifold approximation and projection," J. Open Source Softw. **3**(29), 861–30 (2018).

Downloaded from http://pubs.aip.org/aip/jcp/article-pdf/doi/10.1063/5.0142797/16814539/144109_1_5.0142797.pdf

© Author(s) 2023

[17]G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," Science **313**, 504–507 (2006).
[18]W. Chen, A. R. Tan, and A. L. Ferguson, "Collective variable discovery and enhanced sampling using autoencoders: Innovations in network architecture and error function design," J. Chem. Phys. **149**, 072312 (2018).
[19]C. Wehmeyer and F. Noé, "Time-lagged autoencoders: Deep learning of slow collective variables for molecular kinetics," J. Chem. Phys. **148**, 241703 (2018).
[20]C. X. Hernández, H. K. Wayment-Steele, M. M. Sultan, B. E. Husic, and V. S. Pande, "Variational encoding of complex dynamics," Phys. Rev. E **97**, 062412 (2018).
[21]Y. Bozkurt Varolgüneş, T. Bereau, and J. F. Rudzinski, "Interpretable embeddings from molecular simulations using Gaussian mixture variational autoencoders," Mach. Learn.: Sci. Technol. **1**, 015012 (2020).
[22]J. B. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceeding of 5th Berkeley Symposium on Mathematical Statistics and Probability*, edited by L. M. L. Cam and J. Neyman (University of California Press, 1967), Vol. 1, pp. 281–297.
[23]D. Arthur and S. Vassilvitskii, "K-means++: The advantages of careful seeding," in *Proceeding of 8th ACM-SIAM Symposium on Discrete Algorithms, SODA '07* (SIAM, Philadelphia, PA, 2007), pp. 1027–1035.
[24]J. Shi and J. Malik, "Normalized cuts and image segmentation," IEEE Trans. Pattern Anal. Mach. Intell. **22**, 888–905 (2000).
[25]M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of 2nd International Conference on KDDM, KDD'96* (AAAI Press, 1996), pp. 226–231.
[26]A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," Science **344**, 1492–1496 (2014).
[27]O. Lemke and B. G. Keller, "Common nearest neighbor clustering—A benchmark," Algorithms **11**, 19 (2018).
[28]X. Daura, K. Gademann, B. Jaun, D. Seebach, W. F. van Gunsteren, and A. E. Mark, "Peptide folding: When simulation meets experiment," Angew. Chem., Int. Ed. **38**, 236–240 (1999).
[29]A. Mardt, L. Pasquali, H. Wu, and F. Noé, "VAMPnets for deep learning of molecular kinetics," Nat. Commun. **9**, 5 (2018).
[30]A. Glielmo, B. E. Husic, A. Rodriguez, C. Clementi, F. Noé, and A. Laio, "Unsupervised learning methods for molecular simulation data," Chem. Rev. **121**, 9722–9758 (2021).
[31]W. Brehm and K. Diederichs, "Breaking the indexing ambiguity in serial crystallography," Acta Crystallogr., Sect. D: Biol. Crystallogr. **70**, 101–109 (2014).
[32]K. Diederichs, "Dissecting random and systematic differences between noisy composite data sets," Acta Crystallogr., Sect. D: Struct. Biol. **73**, 286–293 (2017).
[33]T. Lemke and C. Peter, "EncoderMap: Dimensionality reduction and generation of molecule conformations," J. Chem. Theory Comput. **15**, 1209–1215 (2019).
[34]R. J. G. B. Campello, D. Moulavi, A. Zimek, and J. Sander, "Hierarchical density estimates for data clustering, visualization, and outlier detection," ACM Trans. Knowl. Discovery Data **10**, 1–51 (2015).
[35]D. C. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization," Math. Program. **45**, 503–528 (1989).
[36]R. J. Gildea and G. Winter, "Determination of Patterson group symmetry from sparse multi-crystal data sets in the presence of an indexing ambiguity," Acta Crystallogr., Sect. D: Struct. Biol. **74**, 405–410 (2018).
[37]G. M. Assmann, M. Wang, and K. Diederichs, "Making a difference in multi-data-set crystallography: Simple and deterministic data-scaling/selection methods," Acta Crystallogr., Sect. D: Struct. Biol. **76**, 636–652 (2020).
[38]K. Su, O. Mayans, K. Diederichs, and J. R. Fleming, "Pairwise sequence similarity mapping with PaSiMap: Reclassification of immunoglobulin domains from titin as case study," Comput. Struct. Biotechnol. J **20**, 5409–5419 (2022).
[39]L. McInnes, J. Healy, and S. Astels, "hdbscan: Hierarchical density based clustering," J. Open Source Software **2**, 205 (2017).
[40]Prim's algorithm," in *Encyclopedia of Operations Research and Management Science*, edited by S. I. Gass and M. C. Fu (Springer US, Boston, MA, 2013), pp. 1160.
[41]By the average internal cluster RMSD, we mean the average RMSD of all conformations to the cluster centroid.
[42]D. Van Der Spoel, E. Lindahl, B. Hess, G. Groenhof, A. E. Mark, and H. J. C. Berendsen, "GROMACS: Fast, flexible, and free," J. Comput. Chem. **26**, 1701–1718 (2005).
[43]O. Kukharenko, K. Sawade, J. Steuer, and C. Peter, "Using dimensionality reduction to systematically expand conformational sampling of intrinsically disordered peptides," J. Chem. Theory Comput. **12**, 4726–4734 (2016).
[44]E. Chiavazzo, R. Covino, R. R. Coifman, C. W. Gear, A. S. Georgiou, G. Hummer, and I. G. Kevrekidis, "Intrinsic map dynamics exploration for uncharted effective free-energy landscapes," Proc. Natl. Acad. Sci. U. S. A **114**, E5494–E5503 (2017).
[45]K. Lindorff-Larsen, S. Piana, R. O. Dror, and D. E. Shaw, "How fast-folding proteins fold," Science **334**, 517–520 (2011).
[46]We used the trajectories 0, 2, and 3 according to the nomenclature of Ref. 45. We have not used trajectory 1 because the topology file for this specific trajectory differs slightly from the other three in terms of the order and the numbering of the atoms. This issue has also been reported by other researchers.[53]
[47]C. R. Schwantes and V. S. Pande, "Improvements in Markov state model construction reveal many non-native interactions in the folding of NTL9," J. Chem. Theory Comput. **9**, 2000–2009 (2013).
[48]N.-j. Deng, W. Dai, and R. M. Levy, "How kinetics within the unfolded state affects protein folding: An analysis based on Markov state models and an ultra-long md trajectory," J. Phys. Chem. B **117**, 12787–12799 (2013).
[49]M. Ghorbani, S. Prasad, J. B. Klauda, and B. R. Brooks, "Variational embedding of protein folding simulations using Gaussian mixture variational autoencoders," J. Chem. Phys. **155**, 194108 (2021).
[50]H. Sidky, W. Chen, and A. L. Ferguson, "High-resolution Markov state models for the dynamics of trp-cage miniprotein constructed over slow folding modes identified by state-free reversible VAMPnets," J. Phys. Chem. B **123**, 7999–8009 (2019).
[51]P. Deuflhard and M. Weber, "Robust Perron cluster analysis in conformation dynamics," Linear Algebra Appl. **398**, 161–184 (2005), Special Issue on Matrices and Mathematical Biology.
[52]O. Lemke and B. G. Keller, "Density-based cluster algorithms for the identification of core sets," J. Chem. Phys. **145**, 164104 (2016).
[53]E. Suárez, R. P. Wiewiora, C. Wehmeyer, F. Noé, J. D. Chodera, and D. M. Zuckerman, "What Markov state models can and cannot do: Correlation versus path-based observables in protein folding models," J. Chem. Theory Comput. **17**(5), 3119–3133 (2021).

Downloaded from http://pubs.aip.org/aip/jcp/article-pdf/doi/10.1063/5.0142797/16814539/144109_1_5.0142797.pdf

© Author(s) 2023